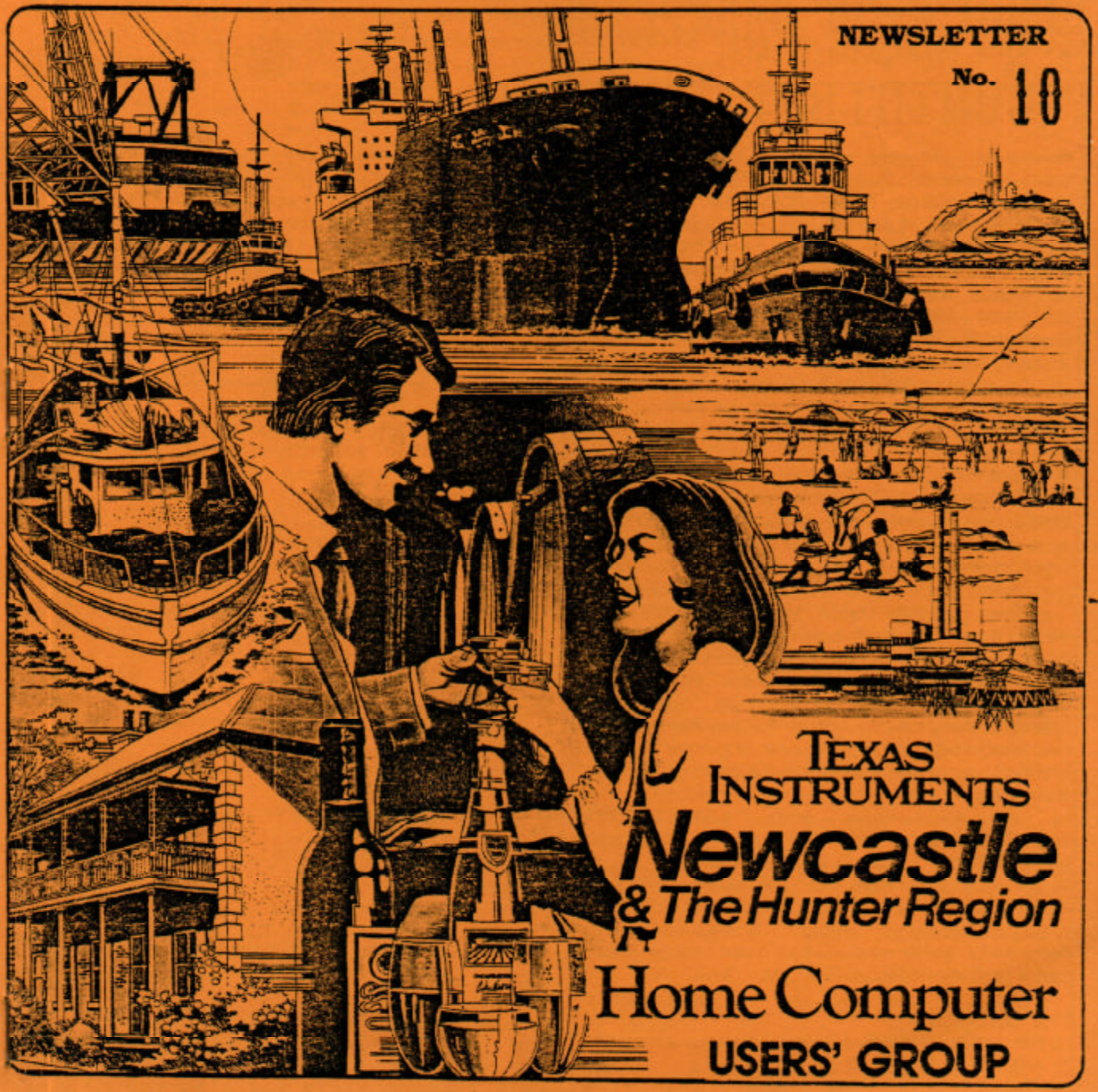
MAY 86

HUNTER VALLEY 99'ERS NEWS

TI 99/4A

HOME COMPUTER NEWSLETTER



NEWSLETTER No. 10

TEXAS INSTRUMENTS

Newcastle & The Hunter Region

Home Computer USERS' GROUP

DISCLAIMER

The HV99 NEWS is the official newsletter of the HUNTER VALLEY NINETY NINE USER GROUP. Whilst every effort is made to ensure the correctness and accuracy of the information contained therin, be it of general, technical, or programming nature, no responsibility can be accepted by HV99 NEWS as a result of applying such information.
TEXAS INSTRUMENTS trademarks, names and logos are all copyright to TEXAS INSTRUMENTS.
HV99 is a non profit group of TI99/4A computer users, not affiliated in any way with TEXAS INSTRUMENTS.

CONTRIBUTIONS

Members and non members are invited to contribute articles for publication in HV99 NEWS.
Any copy intended for publication may be typed, hand written, or submited on tape/disc media as files suitable for use with TI Writer (ie. DIS/FIX 80 or DIS/VAR 80). A suitable Public Domain word processor program will be supplied if required by the club librarian Al Lawrence.
Please include along with your article sufficient information to enable the file to be read by the EDITOR eg. File Name etc.
The preferred format is 35 columns and page length 66 lines, right justified.
All articles printed in HV99 NEWS (unless notified otherwise) are considered to be PUBLIC DOMAIN. Other user groups wishing to reproduce material from HV99 NEWS may feel free to do so as long as the source and author are recognised.
Articles for publication can be submitted to.

THE EDITOR
HV99 NEWS
9 THIRLEMERE PDE.,
TARRO 2322

General address for ALL other club related correspondence.

THE SECRETARY
HV99 USER GROUP
6 ARCOT CLOSE,
TARRO 2322

YOUR COMMITTEE 1986

PRESIDENT
Allen Wright
77 Andrew Rd.,
VALENTINE 2280
Ph. 468120

VICE PRESIDENT
Jim Grimmond
31 Jarrett St.,
TORONTO 2283
Ph. 595751

SECRETARY
Albert Anderson
6 Arcot Close.
TARRO 2322
Ph. 662602

TREASURER
Brian Rutherford
9 Bombala St.,
REDHEAD 2301
Ph. 498184

SOFTWARE LIBRARIAN
Alan Lawrence
35 Bayview St.,
WARNERS BAY 2282
Ph. 436509

PUBLICATIONS LIBRARIAN
Paul Mulvaney
26 Marmong St.,
MARMONG POINT 2284
Ph. 583623

EDITOR
Brian Woods
9 Thirlmere Pde.,
TARRO 2322
Ph. 662307

COMMITTEE
Bob MacClure
75 Deborah St.,
KOTARA SOUTH 2289
Ph. 437431

Garry Jones
53 Janet St.,
JESMOND 2299
Ph. 573744

Tony McGovern
215 Grinsell St.,
KOTARA 2285
Ph. 529162

Peter Coxon
25 Reserve Rd.,
WANGI WANGI 2267
Ph. 751930

SECRETARYS' REPORT

Since the AGM. in March, so many things have, and are happening, that it is hard to take time out to appreciate and thank the people that put so much into the well being of this unique and marvellously active group - HV'99ers. With this in mind I would like to thank everyone concerned as I realise that this support is on-going and freely given throughout our group in true users' group spirit. This, coupled with contact and interchange with other groups, both in Australia and overseas, with the same spirit, produce a nett result that we all benefit and our so called 'shot duck' still performs to unexpected levels for a mere home computer.

At this point I would like to thank Peter Coxon for the manner in which he undertook and maintained the secretary's position as this has made my task much easier to start off with and allows me to duly respect the position of secretary within any oganisation.

With all that, I get to where I should start. Correspondence in and out has been nothing short of hectic this past month, with the most pleasing aspect being that the HV'99ERS influence on the TI user group fraternity is being acknowledged by consistant requests from users all around Australia and the world for membership and newsletter subscription. This to us is reward in itself and provides our group with the type of encouragement needed to maintain support for our common interest - The TI-99/4A.

This gives me an opportune point at which to thank these people for their new interest in our group:-

Ellen Kramer - Ringwood,
New Jersey - USA.
Doug Whalley - Springvale, VIC.
John Piccinich - Caboolture, QLD.
Charles Bagley - Algester, QLD.
Alan Jenkin - Baulkham Hills,NSW.
George Adam - New Lambton (local)

*(and any others that may have gone unmentioned)

Next, the organisation of our groups representation at the Melbourne TI-Fair is well under way with the help and close contact from Peter Gleed and his cohorts in Melbourne. If anyone has suggestions or ideas with regard to the Fair please step forward and say - remember, we are representing YOU!(but do it now as it's only a few weeks away). Along with this our friends in Canberra(CHUG) are looking to rendevous en-route with us on the journey to Melbourne and Charlie Chan has offered his place for a pit-stop on the return journey. Thanks Charlie!

Thanks also go to the Newcastle City Council as they have provided some promotional items to take with us so we can let the rest of Australia know how great the Hunter Valley really is. I will be in touch with the Lake Macquarie Council with a similar request.

With the steady growth of the group over the past 12 months the cost of postage also followed this trend and in order to assist in keeping these costs to a reasonable figure (and keep the treasurer off my back) an application was made to Australia Post for transmission of our newsletter via Registered Publication Mail. This I am happy to report has been granted and should help us to streamline our mail-out.

Another thing that I should mention with regard to mailing and business through the group is that in an effort to reduce communication errors, both inward and outgoing, which unfortunately are inevitable, I have requested that **ALL** group business be directed through the SECRETARY at the address on the newsletter front, inside page.

The matter of the policy towards users groups and back-up service on the 4A from Texas Instruments, which was brought up at the last general meeting, has been addressed by a formal letter to the Manager of Texas Instruments(NSW) on the 14/4/86, however as yet no reply has been forthcoming nor receipt of our correspondence been acknowledged. This is rather disconcerting to many of us and this matter will be followed up in due course.

That's just about it for this report and in closing could I re-iterate one of the groups

policies (probably the most important). Everyone (local, interstate and overseas) is invited to contribute to the HV'99ERS newsletter as this group has as it's foundation, the REAL user group premis, as distinct from the commercial user group.

In the words of a verse of Desiderata: and listen to others, even the dull and ignorant, they too have their story.

Albert Anderson - 4A4Me

A NEW SERVICE FOR MEMBERS

The Group is offering a new service to both buyers and sellers of TI related equipment. Over the last couple of months, an advertisement as been placed in the Newcastle Morning Herald asking anyone who wishes to sell any TI gear to ring Peter Coxon. The aim of this exercise is to establish a "pool" of second hand goods that may be of use to those who are carrying on with the TI.

Peter keeps a list of the equipment available, so if you are interested in buying anything from a back up computer to peripherals or modules, get in touch with Peter. His phone number is on the inside front cover of the magazine.

If you have anything for sale, or know of someone who has, again, Peter will be glad to hear from you.

FILES AND PROGRAMS

This question and answer appeared in Compute!, October, 1985.

What is a file, and what is the difference between a program and a file? When I try to erase a program on disk, my disk drive sometimes gives me a FILE NOT FOUND message.

A file is any collection of data (information) stored permanently on disk or tape, or temporarily in the computer's memory. In many cases, a computer file is the electronic equivalent of a manila file folder: It stores data you create with a computer. Word processing programs store words in files, spreadsheets make files containing numbers, and so on. In that sense, a file may seem very different from a program, which is a set of instructions the computer can load into memory and run. But programs are just a special kind of data - letters, numbers, and other symbols arranged in a pattern the computer understands. Thus, a program stored on disk is a file containing computer instructions rather than some other kind of data. When the disk drive signals FILE NOT FOUND, file is used in a general sense that includes programs along with other kinds of data.

Although these broad definitions apply to all computers, be alert for additional, narrower meanings that apply only to your system or in specific situations. For instance, opening a file to a printer usually means you are opening a communications channel to that device. When in doubt, consult the user's guide for your equipment and pay close attention to the context in which the word is used.

Letters

Dear Sir,

SPECIAL INTEREST GROUPS

Many thanks to the various 'experts' who supported our two workshop sessions, the latter being on Tuesday 22 April at Warners Bay High School.

Special thanks to Brian Woods, a Funlwriter specialist, for his presentations and to Paul Mulvaney for standing in at the last session.

We look forward to many more.
Yours sincerely,
Tim Watkins,
Co-ordinator, SIGS.

DIGRESSIONS

RON KLEINSCHAFER HV99ERS

As many of the HV99 club members know, I live in a fairly remote area mining for Opal as a living. So spare a minute and see if you can help me with a dilemma. It all started several months ago when I decided to try my luck at a place some six miles from where I was working. I attempted to arrange a drilling rig to go to the new area and sink a hole for me. You know what a hole is? It's a hollow whose surface dimensions can vary enormously and always points downward, ie generally towards the centre of gravity if it is to be a hole of any decent stature. Now unable to get the drilling rig to come to the area I wished to work in, a fellow miner and myself decided that the next best thing would be to shift my old hole to the new area. This is where the 99/4A started to earn its keep. Computations were required to determine the volume of rock strata to be removed from around it so that it could be moved. This execise turned out to be more complicated than first realised. It soon became apparent that it would require the programming expertise of a T. McG. Assembly was out of the question as I was not proficient enough, and Extended Basic did not have enough commands to accomplish the task, so after much software restructuring of the XB module, new XB commands were added, some of these being, IF...THEN...SO WHAT, where all results are ignored!! and the most powerful of all, ON ERROR DISREGARD!! where a conclusion is arrived at even if the parameters are wrong. Armed with Earth Resistivity metering equipment, the 99/4A, and a various assortment of digging equipment, we proceeded to the site were the hole was located and carried out our computations. One thing I can say about the old 99/4A is that its accuracy is astounding because using the data printed out we had no trouble digging around the hole and removing it. Now this hole was some 40 ft. deep and without the benefit of lighting was also very dark, you might say like some sort of BLACK HOLE. Anyway we eventually removed it and loaded it onto a truck. Now the track leading to where we wanted to go was to say the least, very rough, then DISASTER struck!! Almost there, we hit a very bad rut and the hole fell off the truck. With much anguish and haste we jumped out to see if any damage had been done, when the biggest disaster of all befell us. You see, in our haste to get out of the truck (we left the computer with its newly developed commands, and all our other equipment in it), we forgot to set the handbrake, and to our utter dismay the truck rolled backwards and fell into the hole.

Would some kind soul out there with suitable software for such a situation pleas help, because you see we can't find the truck or the hole. I told you that it was a BLACK HOLE.

All this happened on the 1st April, 1986.

R.K.

LIFE AT "FUNNELWEB FARM"

BY MRS VAL MCGOVERN

I was asked by Steve Taylor to write a few words on my attitude to Tony's computing - on a good day. In a word - schizophrenia, swinging from being happy at seeing him deeply involved in a hobby from which he obviously gets a lot of fun and satisfaction, to considering citing the T.I. as co-respondent in a divorce action when he's still tapping away at 2am for the third of fourth night in a row.

Part of my problem of course is that I've resisted being bitten by the bug. I figure it's bad enough already having Will & Tony manoeuvering for time on the machine - the race to get up the stairs to claim the machine after dinner or even the problem of prying the possessor loose to come down to meals, without adding one more to the queue, though - sneaky thought! - if I turned into a computer addict too, they might have to cook the meal themselves to tempt me off the machine. Yes, I can see definite possibilities there.

Life here at Funnelweb Farm has changed in other ways with the advent of the T.I. Friends drop by or call at all hours of the day and night, which is fun. The postbox is running hot with letters and disks from all over the world, commenting on Funnelwriter, and filling us in on T.I. happenings in their neck of the woods. All of these letters have to be answered at length with great interest, so the grass gets longer and the odd jobs get put off, but what the hell!

To get back to my reaction to computing, one of the problems which has so far discouraged me from launching myself into the world of computers is a sad lack of understanding of the fact that what constitutes fun and enjoyment to the computer devotee, may seem more like a form of masochism to us, the uninitiated.

For instance, FUN is:-

1) Having a program you've been working on for hours wiped out by a power fluctuation when you have forgotten to save any of it.

2) Accidentally wiping that particularly elegant program you'd almost finished by hitting the wrong key.

3) Having someone turn off the computer with your program in it when you just went out of the room for a few minutes.

4) Mailing off half a dozen disks of FUNNELWRITER to correspondents all over the world half an hour before your son announces with great glee that he's just found a new bug which crashes the whole thing.

Once I have convinced myself of this, and that the muttered curses, and what sounds suspiciously like shouts of agony and frustration coming from the computer room are just figments of my imagination, then move over 99'ers, here I come!

TI-FAIR

As you have probably heard by now, the Melbourne Users Group has organised the first Australian TI-99/4A Users Fair, to be held at the Malvern Town Hall on Saturday, 14th June. There will be on display products from various manufacturers, including Brother Printers, T.I.(!), SENDATA (modems), TELECOM to name a few, as well as stands by various Users Groups from Australia & overseas.

Our Group has been invited to attend and to conduct a stand at the show. At the moment 5 of our members are attending, representing our Group. We aim to have some exciting things to demonstrate, showing to all in attendance that the TI is alive and living in the Hunter Valley. Some of the items we hope to show are:-

1) A P.E. Unit, developed by our own Ron Kleinschafer. We hope that we will have a working model to take.

2) The "Terry Ross" 6 module capacity Cartridge Expander.

3) The "Gary Jones" 32K Memory Expansion.

4) At the Fair, we will premier the lastest, super-dooper, Version 3.3 FUNLWRITER. Although it would appear hard to improve on this fine product from Funnelweb Farm, I am lead to believe this is really something else.

5) Back copies of our Newsletters, so that those poor ignorant souls who have missed it in the past may see what TI has going for it in the Hunter. Our special issue, "Guide to TI-WRITER" will be there as well as the complete McGovern Extended Basic Tutorials.

6) Copies of the programs written by members of the Group will be on sale.

All Groups have been promised access to a Modem, so if arrangements can be made with some of our members who have Modems we may be able to communicate from the Fair. As you can see we will be taking plenty with us to show how a Group like ours can produce material that is world class.

CHEP CHAT

CorComp have announced a new product in the U.S. called the 99Home Sentry. Built in co-ordination with X-10 USA, Inc., it can be used to turn lights on or off, dim or brighten them, turn your stereo on or off and even heat the morning coffee. After the user establishes timings for his appliances, the program is stored in the "X-10 Powerhouse" stand alone RS232 interface, which works with small modules which control the appliances. The package should retail for about US$100, excluding the modules. For further info contact:-
CorComp Inc., 2211-G Winston Rd., Anaheim, CA 92806

****

I reprint, without comment (well nearly), part of the Secretary's Notes from the April issue of the Sydney News Digest, the newsletter of the Sydney Users Group...

" It was recommended that with our change to a company the name of the organisation be changed to reflect the *National status of our group* now that we attract members from all states and territories. The *Australian National TI Club* has been suggested."

* Italics mine - Ed.

****

Using XB & Memory Expansion, try this on for SIZE:-

```
SIZE
CALL INIT
CALL LOAD(-31866,33,0)
SIZE
```

****

THIS ARTICLE FORMED PART OF A DEMONSTRATION BY PAUL MULVANEY

FUNNELWRITER DEMONSTRATION.

COMMAND MODE — FUNCTION 9.

The command mode is the first screen you encounter on entering the editor section of Funnelwriter. The commands are:- Edit,Tabs,Files,Lines,SearcH,RecoverEdit or Quit.

~~~~~~

E

Edit. The capital letters are the commands you use, the first being E for edit which takes you from the Command Mode into the Edit Mode.

~~~~~~

T

Tabs. T for tabs allows you to set up L for left margin, I for paragraph indent, T for any additional tabs required for setting up tables or columns, and R for right margin. If the Formatter is used the tabs can be ignored as they can be set with format commands.

~~~~~~

F

Files. F for files displays a subset of commands. LoadFile,SaveFile,CopyFile,PrintFile,DeleteFile,Purge or ShowDirectory.

LF

LoadFile gets a file or part of a file previously stored on diskette.
To load a file enter DSK*.FILENAME
To load part of a file enter the start line, finish line then the filename
20 66 DSK*.FILENAME

Merging. Whole files or parts of files can be merged with text already in the text buffer. Enter the line that you want the file to come AFTER then the filename. E loads the file after the last line and 0 puts it before the first line of the buffer.
25 DSK*.FILENAME
To merge part of a file type the line after which the file is to be merged, the first line then the last line to be merged and the filename.
0 7 24 DSK*.

SF

Saving. Whole files or parts of files may be saved onto diskette. If the last line of the text buffer is saved the Tabs settings are also saved.
DSK*.FILENAME
205 E DSK*.FILENAME
(If the file already exists on diskette it will be overwritten)

PF

Printing. Whole files or parts of files may be printed. Printing a whole file is normally an automatic process as the printer command is pre-loaded. To set up the default for the printed break (FUNCTION 4) the LOAD program and enter your printer commands in line 130, eg
~~~~~~

RS232.BA=1200 or PIO. To print part of a file enter the start line and end line before the printer command.
1 45 RS232.BA=1200
A PrintF command can be aborted using FUNCTION 4.

DF
Deleting. This command deletes a file from the diskette.
WARNING The file is permanently erased and cannot be recovered.
DSK*.FILENAME

P
Purge. This command clears the contents of the screen and the text buffer and returns you to the Edit Mode. When Purge is selected you are if you are sure (Yes or No). See RecoverEdit.

SD
ShowDirectory. this command catalogs a diskette on the screen. It may be used at any time without affecting the text in the editor.

~~~~~~

L
Lines. When this command is selected a subset is displayed.
Move,Copy,Delete or Show lines.

M
Move allows one line or a consecutive block of lines to be moved to another place in the text buffer and renumbers the lines. Enter the first line, last line and the line after which it is to be inserted.
5 5 41

C
Copying allows one line or a consecutive block of lines to be copied to another location in the text buffer and leaves the original text in its original position. Enter first line, last line and line after which it is to be inserted.
302 E 44

D
Delete erases a block of lines from the text buffer. Enter the start line and the stop line.
WARNING. The deleted lines cannot be recovered.
11 19

S
Show locates a line and displays it as the top line on the screen. Use Ø to display the first line and E to display the last line.
258

~~~~~~

SH
SearcH. When this command is selected a subset is displayed.
FindString or ReplaceString.

FS
FindString allows you to search the text buffer for a character, word or phrase. When found the cursor is positioned on the first character of the string and the line is positioned at the top of the screen. The search takes place from the current cursor position, therefore if you want to search the whole file you must position the cursor on line one left corner. Line numbers may be used to localise the search area.
/string or phrase/
15 58 /string or phrase/

RS
ReplaceString finds and replaces a character, word or phrase (old string) with a new character, word or phrase (new string) in any or all

CONT

places in which the old string occurs. The search starts from the current cursor position. When the search is initiated the prompt appears: Replace String (Yes,No,All,Stop)? When the first string is located Y replaces the string and searches for the next occurence. N searches for the next string without replacing. A replaces all the remaining instances. S returns you to the Edit Mode. Line numbers can be used to localise the search.

```
/frase/phrase/
25 70 /key/console keyboard/
```

RE

RecoverEdit allows all but the first line of a purged file to be recovered. The file cannot be recovered if you have typed any text, loaded a file, turned off the console, run a utility program or used the Formatter.

Q

Quit display the terminating options in a subset.
Save, Purge, or Exit. Save and Purge have already been discussed. Exit takes you back to the options screen and allows you to select Formatter, DM1000 etc. Exit is the only command that cannot be accessed directly from the Command Mode prompt line.

SUMMARY.

E Edit

T Tabs

LF LoadFile

SF SaveFile

PF PrintFile

DF DeleteFile

P Purge

SD ShowDirectory

M Move

C Copy

D Delete

S Show

FS FindString

RS ReplaceString

RE RecoverEdit

Q Quit

E Exit

IT PAYS TO BELONG

It's almost that time of the year again - subscription renewal time. Maybe you wonder if it's worthwhile re-joining because you aren't using your computer as much as you used to, maybe you feel you are not getting enough out of the Group. For whatever reason you are undecided whether to join up again or do what thousands around the world have done - put your trusty TI in the cupboard and forget it.

Well, naturally, the Committee would like you to re-join, and continue using your computer. The purpose of any Users Group is to band together and help others in the Group get the best from their machine. To this end we have in the past endeavoured to provide something of interest in the Group for all members.

Some of the advantages of belonging to the Hunter Valley 99ers must include:-

1) The monthly General Meeting is a time when we can all meet for demos, discussions and general gossiping, swapping tales of monsterous bugs, trying new programs and generally having a good time.

2) A quality monthly Newsletter, devoted to the advancement of the TI-99/4A Home Computer. The newsletter consists of mainly original material, with a smattering of material reprinted from the newsletters of other Groups.

3) The Groups members include people with a wide diversity of experience and knowledge, so any problem you may encounter regarding any facet of using your computer would more than likely be solved within the Group.

4) A vast library of Public Domain software, ranging from games, educational material and utilities to suit any requirement, available to members for the price of the medium. Naturally, no-one's library would be complete without FUNLWRITER, the world acclaimed Word Processor written by our own Life Member, Tony McGovern and his son Will.

5) Technical improvements provided by members of our Group have created widespread interest, for example, the Terry Ross Cartridge Expander, the Ron Kleinschafer P.E. Unit and the Gary Jones 32K Memory Expansion.

6) A comprehensive library of books on all facets of computer use, including technical manuals, the various languages available for the TI, programs to type in, etc., available on loan to members free of charge.

7) Our Group exchanges newsletters with other TI Users Groups in Australia & around the world, and these newsletters are available on loan to members, again free of charge.

8) We have in the last few months begun a Module library that is steadily growing in size. These are available to members free of charge.

9) The Group has its own printer and CorComp Expansion System this too being available free of charge for a period of two weeks at a time to any member.

10) Our Group conducts various Tutorial Groups. Currently there are regular Tutorials in BASIC & Extended BASIC, and in the near future, Assembly Language. Special Interest Groups also have discussed FUNLWRITER and its use, with further groups being organised.

As you can see there are quite a few benefits to be obtained by membership to our Group, so please, RENEW YOUR MEMBERSHIP, you will find the $20 subscription more than worth it.

END

SOFTWARE DEVELOPMENT

Software development or the writing of programs can be divided into several stages. Point number 1 is for me the most difficult, points 2 and 6 the most ignored and point 3 the one requiring the most time.

1. Problem Definition:
Determine what the end product is to be. It includes specifications of inputs and outputs, processing requirements, system constraints (execution time, accuracy, response time) and error handling methods.

2. Program Design:
The design of a program to meet the requirements of the problem definition. Useful techniques include top down design, structured programming, modular programming and flowcharting.

3. Coding:
The translation of the program design into suitable code so the computer can understand it and operate on it.

4. Debugging:
The discovery and correction of coding errors.

5. Testing:
Testing ensures that the program correctly performs the required task.

6. Documentation:
Documentation allows those who must use and maintain it to understand it and so the program can be extended for further applications. Documentation can be in the form of REM statements or printed matter.

REM statements are the most reliable as they are passed on with the program.

7. Maintenance:
The updating and correcting of the program to account for changing conditions.

8. Extension and Redesign:
The extension of the program to solve tasks beyond those described in the initial problem definition. (Usually done by members of User Groups!)

LIBRARIANS REPORT

HI 99'ers,

****Adventurers****

Don't forget the HOTLINE No-583515 with Rodney G.
If anyone would like to have a night of Adventuring(if anyone has time that is,not missing Basic or XBasic classes) let us know and we will try to arrange one.

LUCKY TICKET

The winner this time was Gary Jones. So dont blow a fuse YOU may be LUCKY next time around.

Disk news

I am making up a HV99'er Disk of PD Programes and anyone who would like to contibute,let me have prog.A.S.P. so I can select them to include on it for distribution.
We have now some interesting and Stimulating Simulations sent to us from Melb.U.G.as well as some new Finantial ones to keep your Friendly Bank Manager FRIENDLY and also keep track of Wheeling and Dealing for the TAXMAN coming up soon.

Library Access

Any clubs or indivuals interested in obtaining any PUBLIC DOMAIN software in volume disks have 2 choices
(a) Send blank initialised disks to us with return postage or send us disks with programs on it and we will send at our cost an equal number of disks filled with programs requested or volume disks.
(b) We can supply programs,or volume disks on our disks for the cost of disk and PP(<u>$4.00</u>)

We are not a SHOP and do not sell for profit, any excess is used to buy more material for club usage.
Cassette tapes only to be available at club monthly meetings owing to the problems associated with them.

Your library has about 1500 programs on 110 disks.
Cassette are still a Good buy at the HV99'er monthly meet for $ 3.00.

PETS CORNER

We have received a new Pet called OSCAR from Albert Anderson (sorry Terry its not the Mouse you were looking for)and it is hoped that all of our members will get a chance to look after him and write a few pages on the habits and doings of this animal for our new style Junior Software Column.
At the moment he is being HV Trained by a new member and a report to the ease of this is elsewhere in this issue.

Thats it for now,
Happy programing,
Al Lawrence.

AN EDITORS LAMENT

Getting out a Journal is no picnic;

If we print jokes, we are silly,
If we don't, we are too serious.

If we clip things from other sources,
We are too lazy to write it ourselves.
If we don't we are "stuck" on our own stuff.

If we don't print every last word of all contributions
we just don't appreciate genius.
If we do print them, the columns are filled with junk.

If we make a change in another's article,
we are too critical.
If we don't we are poor Editors.

Now you can bet your life someone will say we swiped this from some other magazine.

You are right - WE DID!

Reprinted from the News Bulletin of the Wentworth Falls Bowling Club!

MINI WORD PROCESSOR DISSECTION

We continue our dissection of the programme at line 250 where we change the screen colour to magenta, just a reminder that you are going into a file read or write routine. then we call SF, which is the save file subprogramme. In the statement CALL SF(L$(),L), we transfer the array L$ which holds the text and the text line counter L. The subprogramme is found at line 800 where to be constant I called the array L$ and the line counter L again. The screen is cleared and a menu is displayed where you have a choice of CS1 or CS2, with our old friend CH used to control the selection, and then depending on the key pressed the variable C$ is assigned "CS1" or "CS2". Next at line 820 the screen is cleared and a message showing how many lines there are to save, and an internal output file of fixed length of 192 bytes is opened to C$ with the statement OPEN #1:C$,INTERNAL,OUTPUT,FIXED 192. Then at line 830 a loop is set up to display the number of the line being saved and then it is printed to the file with the statement PRINT #1:L$(I). After the loop has finished the file is closed with a CLOSE #1 and the SUBEND is performed. The programme then branches back to line 250 and the RETURN statement.

Now at line 260 where we change the screen colour again and the read file subprogramme RF is called, transferring the array L$ and the line counter L to the subprogramme. The subprogramme is found at line 840 where the array and line counter are known by the same name. Then a statement ON WARNING NEXT is used. I used this method of filtering out wrong key presses as it does not

need any conditional statements. I could have used an "ACCEPT AT" with validation for "DIGIT" only, but that still leaves the null string hole for the user to fall in, by inadvertantly pressing the ENTER key before a number. So a simple INPUT with a prompt, and a previous ON WARNING NEXT WORKS well, because if the user presses a letter or just presses the ENTER key the programme branches immediately to the next line that is 850 which goes back to 840. If a valid input is entered then the GOTO 860 is performed where a internal INPUT file of fixed length 192 bytes is opened to CS1. No choice of CS1 or CS2 is given as you cannot read from CS2. A loop is then set up from 1 to the value of L that you input, the line number that is being read is displayed on the screen and the text is read into the computer with the statement INPUT #1;L$(I), next at line 870 the length of that line is subtracted from the amount stored in L$(0), and after the for next loop is finished the file is closed and the SUBEND performed back to the RETURN at 260.

Those last two subprogrammes could probably have been improved by setting the message "Saving line number" and "Reading line number" outside of the loops, but this would have taken more memory as it would have meant an extra line each time.

Back at line 270 where we call the print file subprogramme PF where we have to pass,the array L$ and the line counter L to it. the subprogramme starts on line 880 and the array and line counter are know by the same names again. Next a loop is set up from 1 to L and then the statement PRINT STR$(I);!- prints the line number as a string number and the little squiggles next to it. If I had just had PRINT I;!- the computer would have left a space between the number and the squiggles. Then the colon puts the cursor on the next line and prints the line of text on the screen. Then the loop counter I is checked to see if it is evenly divisible by four and if this is so one of the subprogrammes that we looked at earlier KC is called, which is the press any key to continue subprogramme then the loop is completed with the NEXT statement. The reason for the loop counter is checked is that four lots of five lines of text and their line numbers just fit on the screen, and by calling KP they stay on the screen as long as is necessary to be read. After the loop is completed KC is again used to hold the last screen full of text until the reader has finished. After this the SUBEND is performed back to the RETURN at line 270.

Now at line 280 where the programme branches to if the user pressed 9 in the main menu, the screen turns red to warn you that you have pressed a key that could wipe out your whole file, and a small menu is displayed our old friend CH is called and then if a 1 was pressed L is set back to 0 and L$(0) is assigned "8000" again and a RETURN is performed. If a 2 was pressed the control passes to the next line 290 where you are asked if you have saved your file and then at line 300 a CALL KEY and then a CALL SOUND are called. the CALL KEY to wait for a "Y" or "N" to be pressed and the CALL SOUND to sound a warning tone. If a "N" was pressed then the RETURN is performed back to line 190 and then back to the main menu, but if an "N" was not pressed and nor was a "Y" then the programme loops back to line 300 for you to have another go. If a"Y" was pressed then at line 310 the screen is cleared and the programme stops. I cleared the screen mainly to keep things nice and tidy. Looking at this now it would have probably been better to have allowed a safety out with the purge option as well, also to have assigned all the cells of the array L$ back to nulls. You still have an out if you press 9 by mistake you then press 2 to exit and then press "N" for no you haven't saved your file, and of course you then end up back at the main menu. They are two little extras you could put in for yourself if you need them.

That is the finish of the dissection of Mini Wordprocessor and I could possibly do the same thing with Mini Formatter, but before I do this I would like a little bit of feed back on whether you, the members want this sort of thing or not. in fact does anybody read these programme dissections or learn anything from them.

Brian R.

JUNIOR SOFTWARE REVIEW

"OSCAR"

A Review by Fred Shelley
Hunter Valley 99'ers

A lot easier to say than Optical Scanning Reader: this device reads programs written in bar code, the same code you see on products in the supermarket.

This report on OSCAR has been prepared by a newcomer to the TI-99/4A and computer programming. Being a non typing person, numerous errors occur whilst entering a program from the keyboard, and it is very time consuming, so any device which will reduce both of these is worth investigating.

OSCAR comes with a very informative User's Manual and an introduction sheet which allows you to listen to the various sounds you will hear while using the device. It is important to know these several sounds well, as they convey OSCAR's requirements to the user. Sounds are emitted for:

1) "End if line" - a beep meaning you have scanned the line correctly.
2) "Commence scanning new line" - this beep sometimes follows the end of line beep , and is sometimes delayed whilst OSCAR down loads to the TI.
3) An "error" buzz - meaning the line has to be re-scanned.
4) A "time out" signal if OSCAR has not been used for 30 seconds, and after another 30 seconds OSCAR will power down.

The method of operation is to place template over each page of the desired program and move the sensing wand left to right on line 1 and right to left on line 2. This sequence is followed down to the last line, which is #31, and the process is repeated for the next consecutive pages.

Sounds easy, doesn't it? Well, it isn't - at least not at first attempt.

A flat surface is necessary to scan the pages on, and you should make yourself comfortable as it isn't easy to hold the template and use the scanning wand smoothly. The scanning wand lead tries to bend the pages of the program yet to be scanned unless you take steps to prevent it.

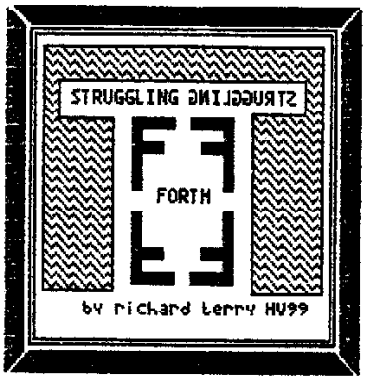
The scanning wand reading tip must be clean and a smooth movement over the bar code is necessary to avoid an error buzz. Once this technique is mastered, using OSCAR is a breeze and programs may be loaded with only the occassional line having to be re-scanned. Some lines are difficult to scan correctly and may need to be re-scanned up to ten times before the correct beep is heard.

As with any system there are a few points to bear in mind:
1) The bar code can be corrupted by the program sheet being creased or dirty, and an error buzz will result.
2) Each line must be entered in the correct order.
3) The scanning head has to be at to the bar code or an incorrect scan will result, and further attempts must be made to satisfy OSCAR's requirements.
4) When finished scanning a program, the wand must be properly located to avoid draining the battery power source. If this is not done then OSCAR will "time out" after 1 minute and it will be necessary to depress the micro switch to activate OSCAR for re-use.

Once suitable scanning techniques have been mastered, OSCAR is quick and easy to use for loading programs. The catch is you will need a bar code printer to convert your own devised programs so OSCAR can read them, otherwise you still have to load them via the keyboard at least once.

I have thoroughly enjoyed using OSCAR, and recommend it to others who would like to enjoy it's programs. But beware! The program "OSCAR's Speller" is for Commodore machines, and will not load into the TI.

END



Deadline once again a few hours to go. Joe maintains I deliberately leave writing these articles till the last minute as I operate best under pressure, pun intended . The reality is really just that I never decide what to write about till the last minute, ie until forced into it by the deadline.

Just for something completely different I thought today, once again breaking last months promise, that we'd take a look at using numbers in programs. I'll talk briefly about double numbers, single numbers and a little more about applications using floating point numbers and their manipulations.

LEARNING BY EXPERIMENTATION.

The following example will show how ones thought on a particular area of FORTH can evolve through the use of experimenting with words, approaches to problems. Indeed to learn Forth you must be sitting at the terminal, writing, examining memory addresses and seeing what the definitions do to them. As the months roll by I am sure I will change many of my approaches, sometimes radically, sometimes subtly.

REFER TO SCR# 64.

Before we start I have modified our string definition of GET$ and MOVE$ This new definition adds an Ascii blank or character 32 to the end of any strings accepted. Well to be perfectly honest, this is done by WORD, which when examining our input deposits the result at HERE with one or more following Ascii blanks. When we wrote these originally we chose to ignore these blanks as being unnecessary, which, if your strings will only ever be non-numeric, they are. However if you try converting a numeric string to a number using the old definition things are different.

Boot your Forth and load our old definition of GET$/MOVE$. Now type :

Ø VARIABLE BUFR 1Ø ALLOT

Within a program we may want to accept numbers as strings in order to validate what we want to be entered eg user may only be allowed to enter money format strings or even particular combination of numbers.After validation this data may then need to be in plain number format, either on the stack, or stored for later use,in a CONSTANT or VARIABLE, by other parts of the program. Hence we need some way of converting our character string to a numeric value on the stack. There are several ways to do this, the first we will examine converts it to a double number on the stack.

The Forth word which does that is NUMBER, and its primitive (NUMBER)

Lets experiment with our old definition. Type in:

BUFR 6 GET$, and enter a number eg 123.44

Now type BUFR NUMBER and press enter. Forth will, rather disconcertingly reply NUMBER?. This is the response we get when a word it not in the dictionary, isn't it. Yet referal to our manual, such that is it, insists the word is "RESIDENT" ie in our core. VLIST your dictionary and sure enough there it is.

What has gone wrong. Well,all this may seem simple to you seasoned Forth users out there laughing at

my ignorance. Would you believe it took me ages to discover it because I was so ?@!**'ed off that it wouldn't work,in desperation I wrote my own definition which would, using the primitive(NUMBER):

```
: CONVERT           ( addr---DN)
          Ø Ø       (DN accumulator)
          ROT 1-
          (NUMBER)
          DROP DROP ;
```

where addr contains ASCII characters deposited by EXPECT, without a decimal point eg

```
BUFR 6 EXPECT 12
```

Typing BUFR CONVERT leaves 12 on the stack

Frustration over, experimentation had prevailed. I had a definition which, although I didn't understand HOW it worked, it did,leaving me free to continue programming.

So to be honest months went by. My tussle with NUMBER, having assumed epic proportions, slowly faded, or more likely was actively repressed into the lower addresses of my cortex. I did't even notice it in the manual any more.

But, such are the twists of fate, it bubbled to the surface once again.I'd occasionally used it, and it seemed to work, albeit inconsistently. I then noticed that it worked initially in situations where I had blanked the accept buffer with Ascii 32's before accepting the data, AND, my buffer was longer than the inputted character string plus count.

Twas then the penny dropped. How stupid I was. The bloody thing needed a
delimiter to tell it where to stop, an ASCII blank.
So that's why WORD leaves blanks after one uses it!!!!! DUMB!!!!!

Lets try again.

```
Type BUFR 7 BLANKS
     BUFR 6 GET$ 123.44
     BUFR NUMBER ->presto:123.44 ok.
```

Examining the definition of NUMBER shows why my cretinous brain had such trouble with it. There is no mention that one needs a delimiter. Typical of the manual. Like most glaringly obvious things in life, it is eminently easy to see using a retrospectoscope.

We can now look at our modified version of GET$ and MOVE$. MOVE$ has been redesigned to add 1 to the character count hence it moves the extra blank ascii 32 byte from HERE to your designated address. We now no longer need to blank buffers in order to convert a string of numbers to a double number on the stack.

SINGLE NUMBERS-PRACTICAL APPROACH

For the technical side refer to a text book such as page 150 onwards in BRODIES STARTING FORTH. For the practical application try this. Forget that cup of coffee, or that its after midnight and your spouse is still begging your company in bed(god knows why,because hacking away here hour after hour your probably not capable of anything), sit before the terminal and type:

```
Ø VARIABLE BUFR
```

This allots a 2 byte buffer. Now try sticking numbers into it using these definitions:

```
: STORE BUFR ! ;
: FETCH BUFR ? ;
```

```
type: 1 STORE ok
      FETCH   1 ok.
```

```
etc with increasing numbers eg
20000 STORE ok
      FETCH 20000 ok.

32767 STORE ok
      FETCH 32767 ok

32769 STORE ok
      FETCH gives wrong answer
```

What we get back is garbage, not what we put in. In other words, for the practical application, forgetting about why, we can only store values up to 32767 as single number. Any value over that cannot be thus stored. For many programs this does not matter, for others such as those requiring money amounts in high figure, or mathematical calculations, some other method of storing is needed.

An aside for a moment.

Type 55 STORE ok.
Now instead of using our definition FETCH type:

```
     BUFR    @ . 55 Ok.
     BUFR 1+ @ . 55 OK.
```

ie with single numbers identical values are stored in two contiguous byte addresses, in this case BUFR and BUFR 1+. As later articles will show this can have practical applications when saving data to disk etc from arrays.

FLOATING HELL.

Floating point maths is one of the other ways we can manipulate larger numbers than 32767. For some reason most Forth devotees I've spoken to seem to shy off this, which is a pity in many ways as it is fairly easy to manipulate once you get the hang of it. Unlike many computer owners we don't have to go to the trouble of writing out own assembler routines to create floating point numbers, as they are already in ROM in the console, and we can access this provided we load the -FLOAT option into our dictionary.

What are the penalties associated with its use. A definite speed disadvantage is noticeable if you compare routines written in both. Mind you, that is for purists as its still very quick.

What are the gains.Well, I'm not quite sure, as I havn't really broached the realms of double numbers yet, but I've found it easy to use in programs where I want to accept input as a string preceded by a count,so as to validate its contents, and later convert to floating point, and re-convert to strings for storage/display.

WHAT ARE FLOATING POINT NUMBERS?

I've no idea. I only use them. Remember this is a non-computing mans guide to Forth. Ask Tony McGovern.

ACCEPTING NUMBERS FOR FLOATING POINT

Referring to the Manual chapter 7 will show you several ways of directly accepting a number as a floating point from keyboard to an address using F! and F@ etc. I havn't bothered using this yet. The only main practical point is you must allow yourself enough room in a buffer to fit the things as they occupy 8 bytes of space eg

Ø VARIABLE FBUF 6 allot would suffice.

Once you have your floating point number, hereafter referred to as F.#, it is manipulated on the stack with similar operators as single numbers FDUP=DUP FSWAP=SWAP etc.

Load SCR# 64 with GET$/MOVE$.
Lets set ourselves a task. To accept a number as a character string, store it as a string, change it to a floating point number, manipulate it , and converting it to a string.

```
Type Ø VARIABLE BUFR 5 ALLOT
     ( allot 7 byte buffer )
     BUFR 5 GET$  12.44
```

Using our new definition of GET$ the buffer will be full with the preceding count and following blank:

```
5-count
49-1
5Ø-2
46-.(decpoint)
52-4
52-4
32-blank
```

Using available commands lets design a word which will convert a string at an address and leave it as a F.# on the stack. Looking up the manual we peruse anything vaguely having anything to do with F.#'s. we stumble across:

VAL:which converts a string at PAD to a F.# and placed into FAC. We groan loudly(not too loudly as remember the wife has just lulled off to sleep-its 2am-trouble coming up in the morning).What and where is FAC? It turns out this is the address of the FAC register. What the hell is that!!!!.

Ok lets go, we scribble

```
BUFR PAD MOVE$
```

we now have string at PAD.

```
VAL
```

now its disappeared into FAC. We really do want it on the stack, so re-checking our definitions for anything with a FAC in it we find:

```
FAC>
```

Eureka you scream. A big mistake actually,as you woke the wife and half the kids. You lock the computer room door and ignore the screaming.

Re-writing it:

```
: $->F.#
          BUFR PAD   ( FROM,TO       )
          MOVE$      ( move to PAD   )
          VAL        ( move to FAC   )
          FAC> ;     ( dump ->stack  )
```

but thats only good from one address so we simply delete BUFR in our final definition(see screen# 61)

Once on the stack we can manipulate it how we like,using commands in the -FLOAT option. To reconvert it from a F.# to a string we find a command STR AND >FAC:

```
: F.#->$   >FAC STR ;
```

STR is supposed to convert a number in the FAC to a string with a count in BASIC format and leave it at PAD. Here I found something strange:

Go through the above steps and leave the String of 12.44 as a F.# on the stack and re-convert to a string using F.#->$
Now examine the contents of PAD using C@:

```
: PAD-DUMP 8 Ø
           DO PAD DUP
              I +
              C@ . CR
           LOOP  ;
```

We find that we get

```
PAD       6=count
PAD 1 + 32=ascii blank
 "  2 + 49=1
 "  3 + 5Ø=2
 "  4 + 46=.
 "  5 + 52=4
 "  6 + 52=4
    7 + whatever.
```

Ie for some reason it sticks an Ascii blank at count +1 and ups the count. I dont't now why. Note byte 7 is labelled whatever. This definition of re-conversion to a string also DOES NOT put a trailing blank back. For my application this does not matter,but it may for yours so you will have to re-adjust MY-STR to do this.Now refer to screens 6Ø-63

I have redefined STR AS MY-STR to simply shift back one byte and delete the blank and adjust the count back. It seems to work ok, but perhaps like our original definition of GET$ the reason may later become apparent to me, unless some of you care to enlighten me.

As I'm running out of time and space I'll include a few more words to help you manipulate balance data from and to a balance array. In my case I have 28 categories of balances each containing up to $99999.99 amounts (1Øbytes space needed for each). BAL-ADR calculates the offset into this array where the balance of each category starts. Perusal of the code on screens 6Ø-63 should be self explanatory. If I do say so myself I feel this is good Forth code. It is self explanatory, the words are clear, precise and do exactly as they say. The code of compact, visually pleasing and well set out. The comments next to them provide a good explanation of each step.Uncharacteristically the falling tickets are clogging up my keyboard so I'll leave you with a parting message.

Hunched over you VDU your visual senses become dimly aware of the first few rays of light penetrating the curtains of your den,your auditory senses catch the splintering of the door panels (you forgot to lock up the axe again-must be your wife) and your taste buds tell you the only resolution to the problem i a beer at the early morning pub on the waterfront. As you clamber out the window, read the following: An introduction to Flow Charting. This will be an invaluable aid to both your future excursions into programming and your hip-pocket nerve at the pub.

ADDRESS FOR CORRESPONDENCE:

R.TERRY.
141 DUDLEY RD.
WHITEBRIDGE 229Ø.
Ø49 436861/2245Ø.

```
UNSWCH
SCR #60
  0 ( B/CHEQUES - redfinining STR   24Mar86)
  1
  2 : MY-STR                        ( MY DEFINITION OF STR COMMAND      )
  3               STR               ( leave string with count at PAD    )
  4               PAD 2+            ( start adress to move bytes        )
  5               PAD 1+            ( destination adress to move bytes  )
  6               PAD C@ 1- DUP     ( fetch and decrement byte count    )
  7               PAD C!            ( re-store to first byte at PAD     )
  8               CMOVE ;           ( move the rest 1 space to left     )
  9
 10 ( NOTE: this was necessary because for some reason FORTH leaves
 11         the count +1 in byte zero at PAD because it inserts an
 12         ASCII blank between count and rest of string           )
 13
 14
 15

SCR #61
  0 ( B/CHEQUES - balance adjusting 24Mar86)     ( CHECKED CLEAN)
  1 : $->F.#                        ( conv string with cnt ->FP# on stk)
  2               PAD MOVE$         ( expects Adress, copy of $ -> PAD  )
  3               VAL FAC> ;        ( leave floating point # on stack   )
  4 : F.#->$                        ( expects floating point # on stack)
  5               >FAC MY-STR ;     ( leaves string with count at PAD   )
  6
  7
  8 : BAL-ADR   CATBALADR @ SWAP        ( expects category #            )
  9             1- 10 * +   ;           ( leaves start adr of categ     )
 10
 11
 12
 13
 14
 15

SCR #62
  0 ( B/CHEQUES - balance adjusting 24Mar86)     ( CHECKED CLEAN)
  1
  2 : @OLD-BAL    CATEG             ( category adress left on stack     )
  3               NUMBER DROP       ( categoy number left on stack      )
  4               BAL-ADR           ( adress offset into balance array  )
  5               DUP               ( 2 copies,one left on stack        )
  6               $->F.#      ;     ( convert balance to F.# on stack   )
  7
  8 : +AMOUNT     AMOUNT            ( adress of current inputted amount)
  9               $->F.#            ( leave amount as F.# on stack      )
 10               F+          ;     ( add,result as F.# on stack        )
 11
 12
 13
 14
 15
```

```
SCR #63
  0 ( B/CHEQUES - balance adjusting 24Mar86)     ( CHECKED CLEAN)
  1 : -AMOUNT     AMOUNT          ( adress of current amount          )
  2               $->F.#          ( leave amount as F.# on stack      )
  3               F-    ;         ( subtract leave as F.# on stack    )
  4
  5 : !NEW-BAL     F.#->$         ( convert F.# on stck to $ as PAD   )
  6               PAD SWAP        ( leave PAD adr,swap categ to top   )
  7               MOVE$  ;        ( move back to balance array        )
  8
  9 : +ADJUST     @OLD-BAL        ( fetch existing amount of balance  )
 10               +AMOUNT         ( add ne amount to this, leave tot  )
 11               !NEW-BAL  ;     ( re-store to balance array         )
 12
 13 : -ADJUST     @OLD-BAL        ( fetch existing amount of balance  )
 14               -AMOUNT         ( decrement this balance            )
 15               !NEW-BAL  ;     ( res-store to balance array        )

SCR #64
  0 ( ABRIDGED CODE - STRING WORDS          )      (  STACK EFFECTS)
  1
  2
  3
  4
  5 : MOVE$ SWAP COUNT 2+ SWAP 1- ROT RU! CMOVE ; ( Adr1 Adr2--- )
  6
  7 : GET$  TIB @ SWAP EXPECT 0 IN ! 13 WORD      ( Adr to put new )
  8         HERE SWAP MOVE$     ;                 ( string,count   )
  9
 10
 11
 12
 13
 14
 15
```

HITCH HIKERS GUIDE TO THE GALAXY

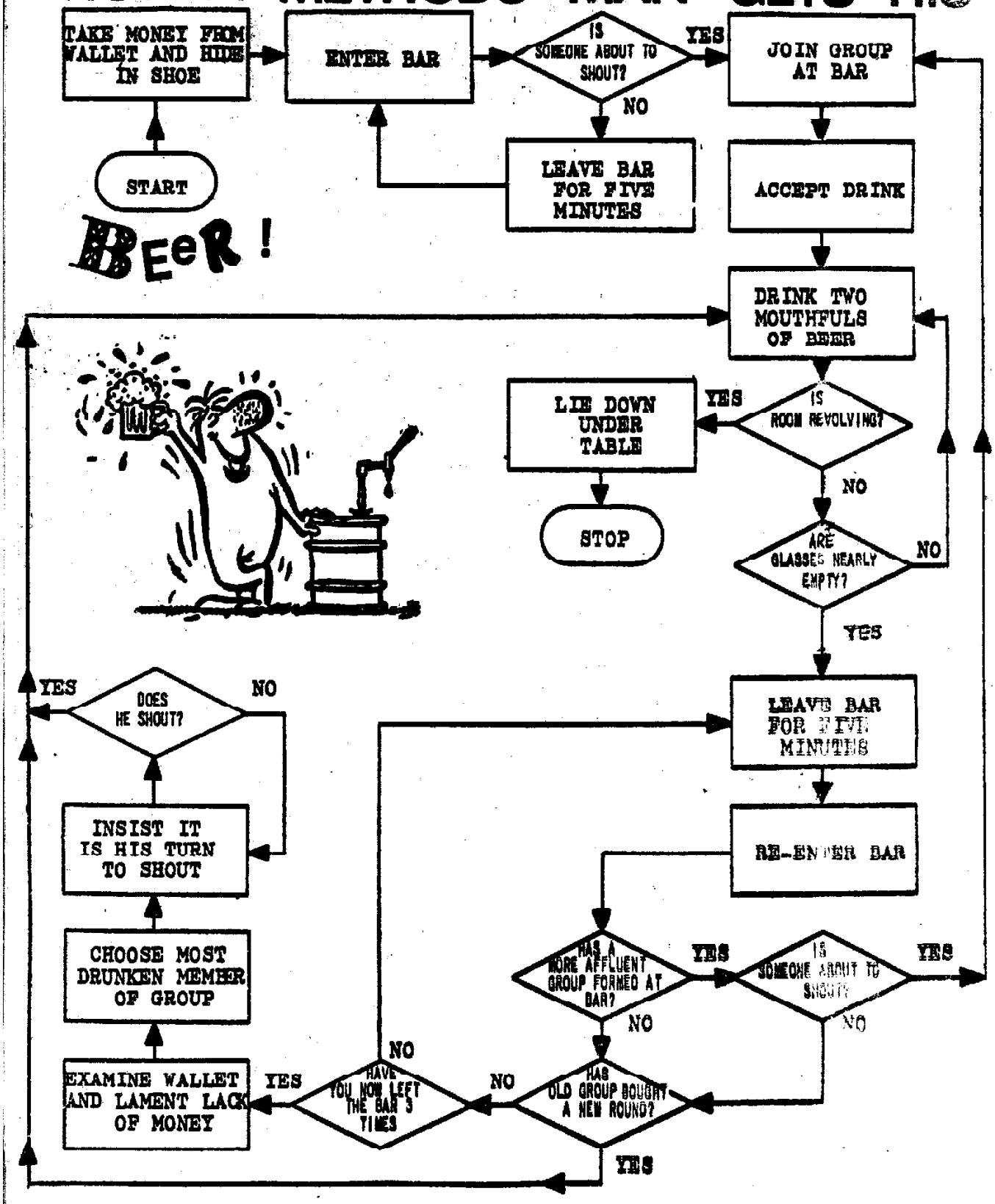
It all started in the late 1970's when the radio series first started. It was so succesful that there was a second series followed by a book then another two books, a stage play and a television series. The road to success was already built. All that was left was to cross the bridge and start. Who else but INFOCOM would dare. When you first get the adventure home and open it, you see a friendly demolition order for your home, a set of extremely useful peril sensitive sunglasses (which you naturally can't see through), a don't panic badge, a piece of fluff, another demolition order, this time for your planet and a small polythene bag (with which to destroy microscopic space fleets). Throughout the adventure you may consult the H.H.G.T.T.G. This is an electronic book all the problems in hitch hikers are logical and have logical solutions (if you are insane). DO NOT THINK THAT THE GAME AND THE BOOK ARE THE SAME EVEN YOUR MAIN OBJECTIVE IS DIFFERENT! INFOCOM has surpassed itself in this game and already there are rumours of hitch hikers two. So remember that if it's 4am it must be hitch-hikers.

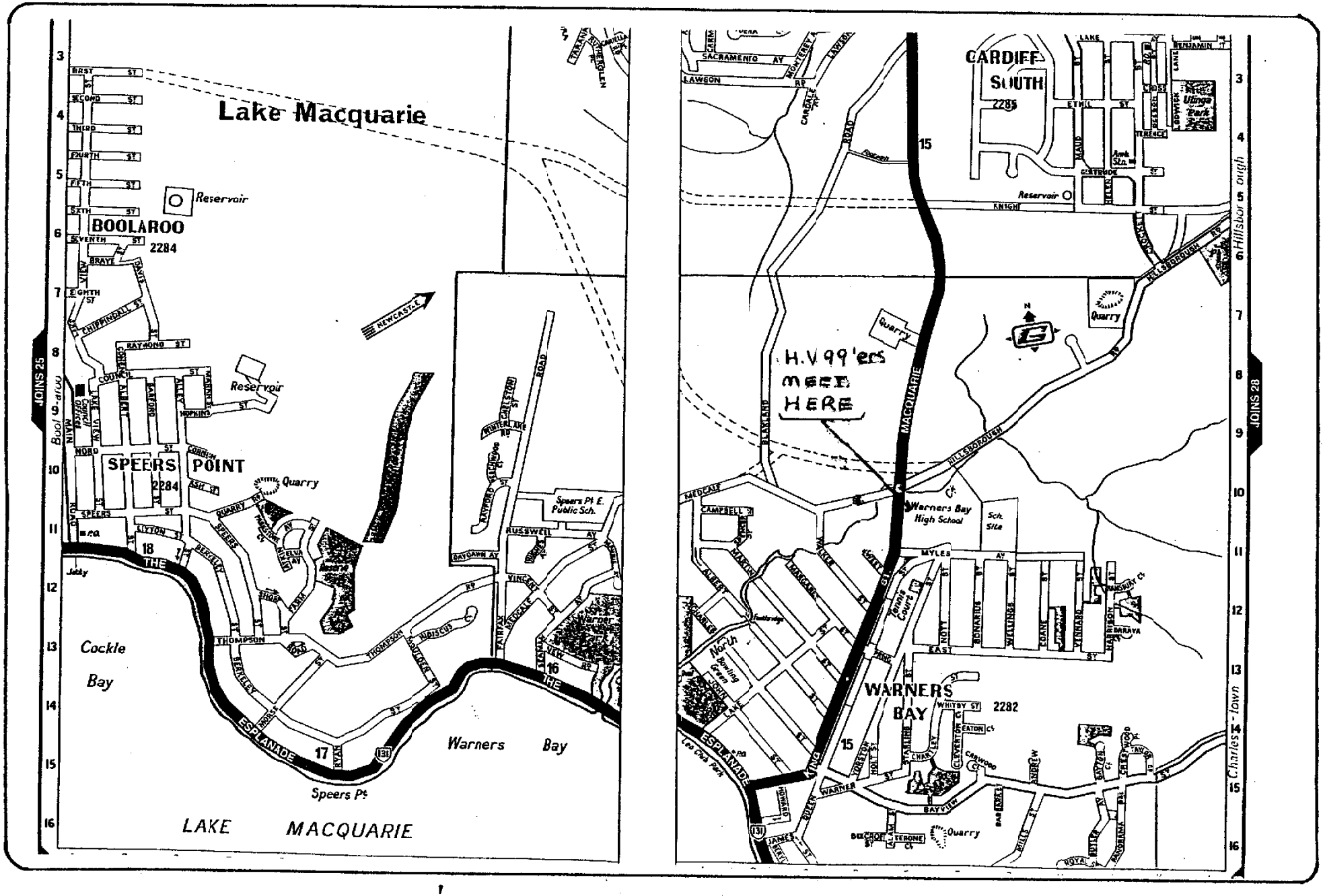
RATING 10/10

RODNET G.

AN INTRODUCTION TO FLOWCHARTS, OR-

HOW A METHODS MAN GETS HIS BEER!




Lake Macquarie
BOOLAROO
2284
SPEERS POINT
2284
Reservoir
Quarry
Speers Pt E. Public Sch.
Cockle Bay
Warners Bay
Speers Pt
LAKE MACQUARIE
CARDIFF SOUTH
2285
WARNERS BAY
2282
H.V 99'ers MEET HERE
Warners Bay High School
Sch. Site
JOINS 25
JOINS 28
Buol Buol-arou
Charles-town
Hillsboro ough

THE INFORMATION PAGE

COMING EVENTS

SIGS meeting: Saturday, 31st May at 1.00pm (see enclosure)
Next Committee meeting: Tuesday, 3rd June at 6.45pm
Next General meeting: Tuesday, 10th June at 7.00pm
1st Australian TI-Fair, Malvern Town Hall (Melbourne): Saturday, 14th June 12 noon to 6.00pm

AGENDA for JUNE GENERAL MEETING

OSCAR demonstration
MULTIPLAN demonstration
Monthly disk/tape demonstration

GARY JONES' EXTENDED BASIC GROUP

Starting Sprites

AL WRIGHT'S BASIC GROUP

Cassette Files
Debugging

All meetings are held at the Warners Bay High School

Annual subscriptions to the Group cover the period 1st July to 30th June following year.

People interested in joining our Group are asked to contact:

The Secretary,
Hunter Valley 99'ers Users Group,
6 Arcot Close,
TARRO. N.S.W. 2322
AUSTRALIA
phone (049)662602

The annual subscription is as follows:
Residents of Australia...$20
Overseas residents.....$40 (Airmail)
$30 (Surface Mail)

Back issues of our Newsletter are available for $1.00 plus postage